# **Front-end Engineer Interview Coding Assignment**

**NAME : Nethravathi GR**

## Index1.html [ HTML5 code] :

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Nested Tags Tree</title>

  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

  <link rel="stylesheet" href="styles1.css">

</head>

<body>

  <div class="container mt-4">

    <div id="tagView"></div>

    <button id="exportButton" class="btn btn-primary mt-3">Export</button>

  </div>

  <script src="script1.js"></script>

</body>

</html>

## Styles1.css [CSS3 code] :

.tag {

         border: 1px solid #ccc;

         margin: 10px;

         padding: 10px;

       }

       .tag-header {

         display: flex;

         align-items: center;

         background-color: #f0f0f0;

         padding: 5px;

       }

       .collapse-btn {

         cursor: pointer;

         margin-right: 5px;

       }

       .tag-name {

         flex-grow: 1;

         border: none;

         background-color: transparent;

         outline: none;

       }

       .add-btn {

         margin-left: auto;

         background-color: green;

         color:whitesmoke;

       }

       .children {

         margin-top: 5px;

       }

       .tag-data {

         margin-top: 5px;

         width: 100%;

         border: none;

         background-color: transparent;

         outline: none;

       }

## Script1.js [JAVASCRIPT code] :

const tree = {

         name: 'root',

         children: [

           {

             name: 'child1',

             children: [

               { name: 'child1-child1', data: "c1-c1 Hello" },

               { name: 'child1-child2', data: "c1-c2 JS" }

             ]

           },

           { name: 'child2', data: "c2 World" }

         ]

       };

       function createTagView(tag) {

         const tagDiv = document.createElement('div');

         tagDiv.classList.add('tag');

         const headerDiv = document.createElement('div');

         headerDiv.classList.add('tag-header');

         const collapseButton = document.createElement('button');

         collapseButton.innerHTML = 'v';

         collapseButton.classList.add('collapse-btn');

         collapseButton.addEventListener('click', () => toggleCollapse(tagDiv));

         const nameInput = document.createElement('input');

         nameInput.value = tag.name;

         nameInput.classList.add('tag-name');

         nameInput.addEventListener('change', () => updateTagName(tag, nameInput.value));

         headerDiv.appendChild(collapseButton);

         headerDiv.appendChild(nameInput);

         if (tag.children) {

           const addButton = document.createElement('button');

           addButton.innerHTML = 'Add Child';

           addButton.classList.add('add-btn');

           addButton.addEventListener('click', () => addChild(tag));

           headerDiv.appendChild(addButton);

           const childrenDiv = document.createElement('div');

           childrenDiv.classList.add('children');

           tag.children.forEach(child => {

             const childTag = createTagView(child);

             childrenDiv.appendChild(childTag);

           });

           tagDiv.appendChild(headerDiv);

           tagDiv.appendChild(childrenDiv);

         } else if (tag.data !== undefined) {

           const dataInput = document.createElement('input');

           dataInput.value = tag.data;

           dataInput.classList.add('tag-data');

           dataInput.addEventListener('change', () => updateTagData(tag, dataInput.value));

           tagDiv.appendChild(headerDiv);

           tagDiv.appendChild(dataInput);

         }

         return tagDiv;

       }

       function toggleCollapse(tagDiv) {

         const collapseButton = tagDiv.querySelector('.collapse-btn');

         const childrenDiv = tagDiv.querySelector('.children');

         const isOpen = childrenDiv.style.display !== 'none';

         childrenDiv.style.display = isOpen ? 'none' : 'block';

         collapseButton.innerHTML = isOpen ? '>' : 'v';

       }

       function updateTagName(tag, newName) {

         tag.name = newName;

       }

       function updateTagData(tag, newData) {

         tag.data = newData;

       }

       function addChild(parentTag) {

         if (!parentTag.children) {

           parentTag.children = [];

         }

         const newChild = {

           name: 'New Child',

           data: 'Data'

         };

         parentTag.children.push(newChild);

         renderTagView();

       }

       function renderTagView() {

         const tagView = document.getElementById('tagView');

         tagView.innerHTML = '';

         const rootTag = createTagView(tree);

         tagView.appendChild(rootTag);

       }

       document.getElementById('exportButton').addEventListener('click', () => {

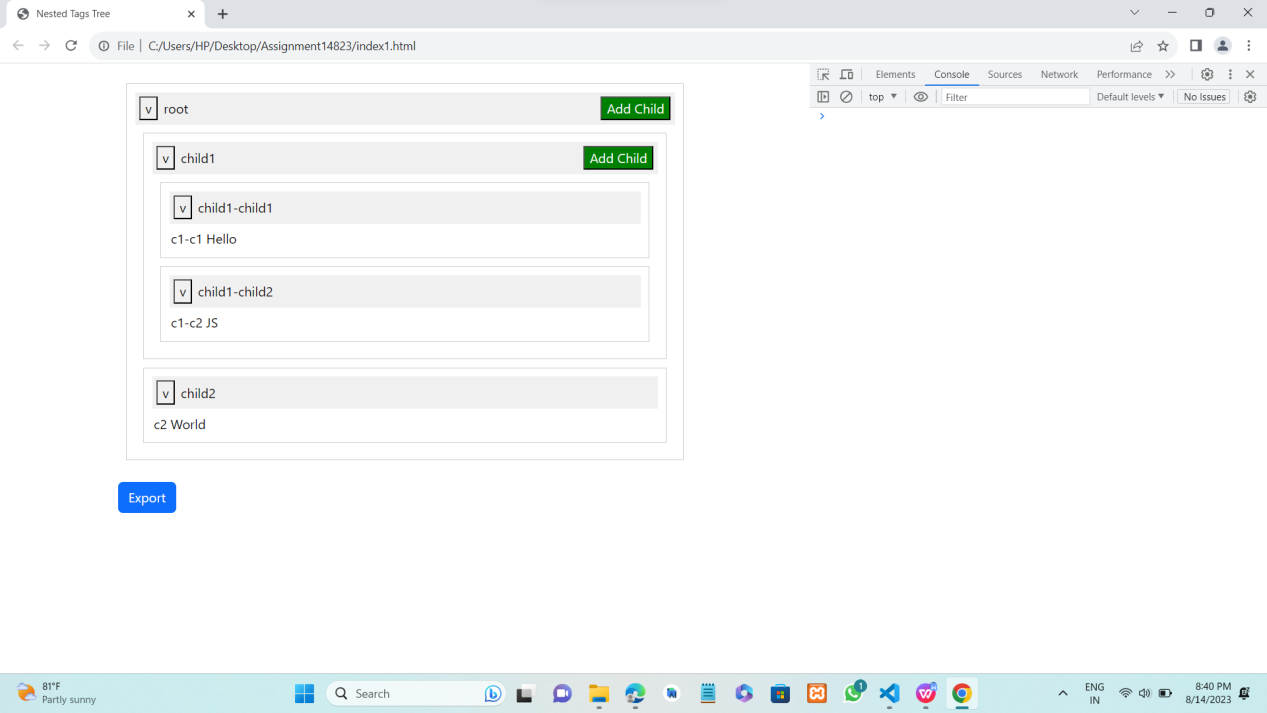
         const exportedData = JSON.stringify(tree, ['name', 'children', 'data'], 2);

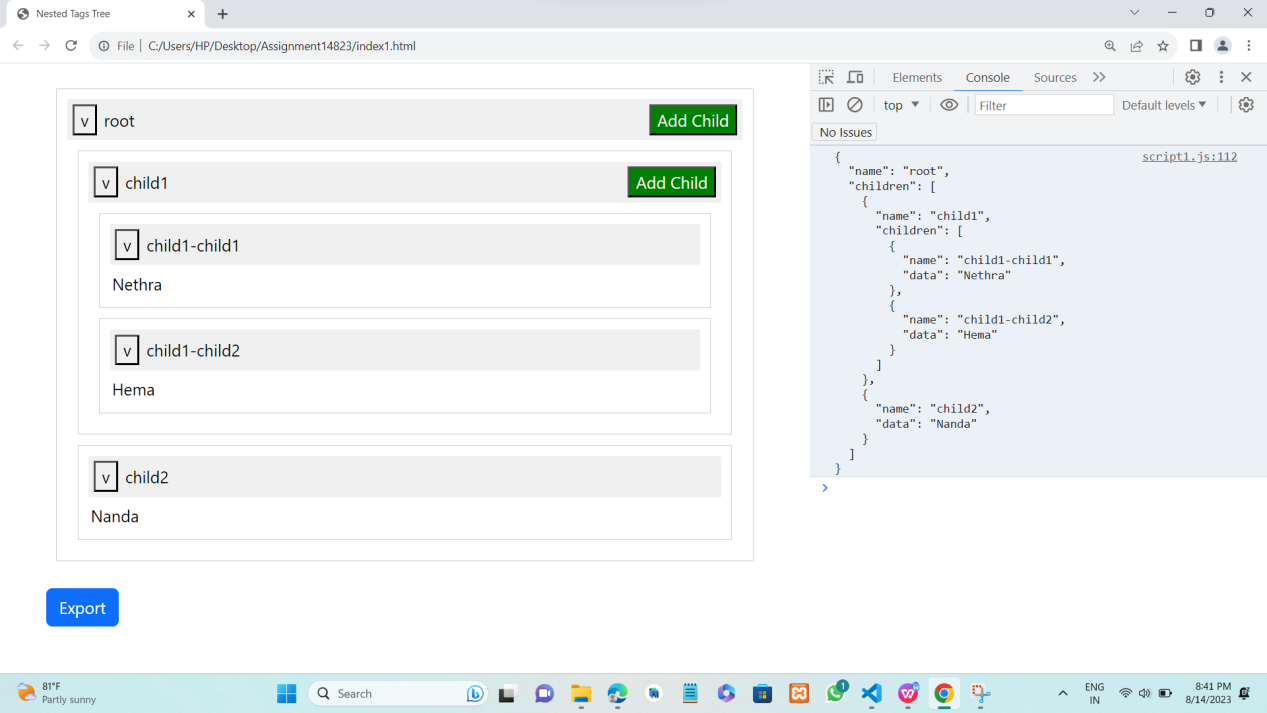
         console.log(exportedData);

       });

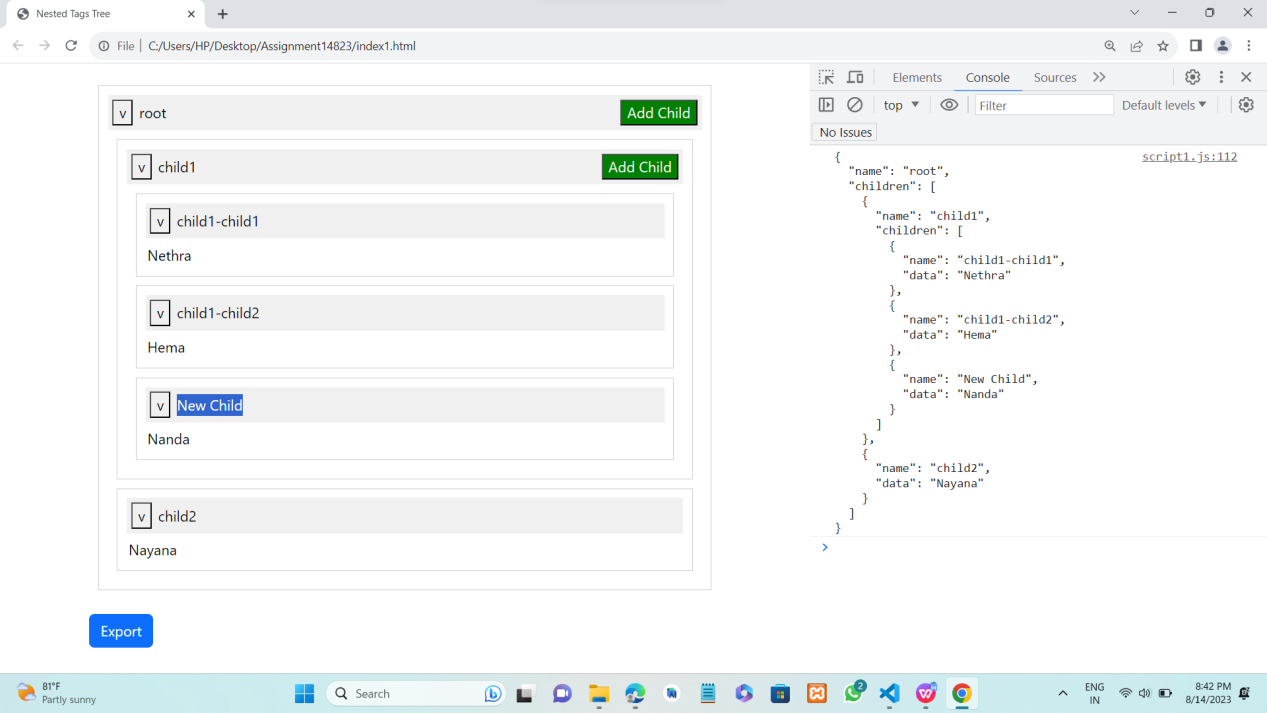
       renderTagView();

## OUTPUT :

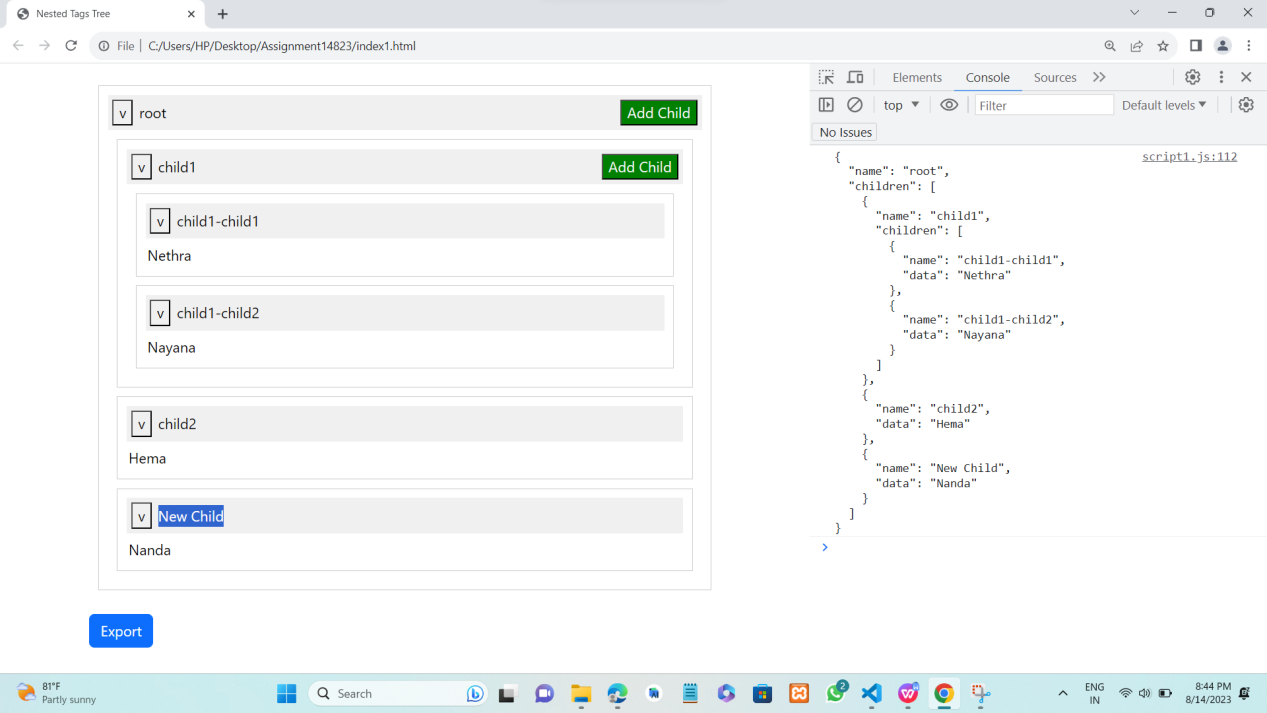




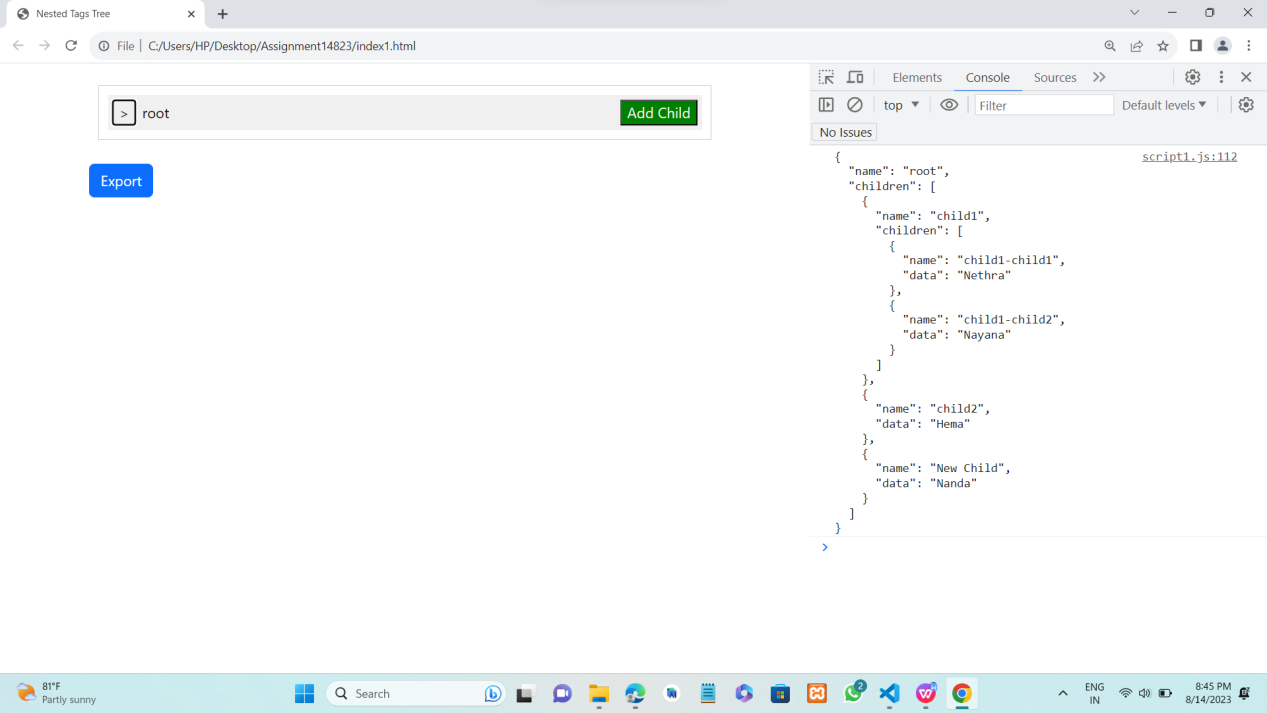
## Adding new child for “child1”



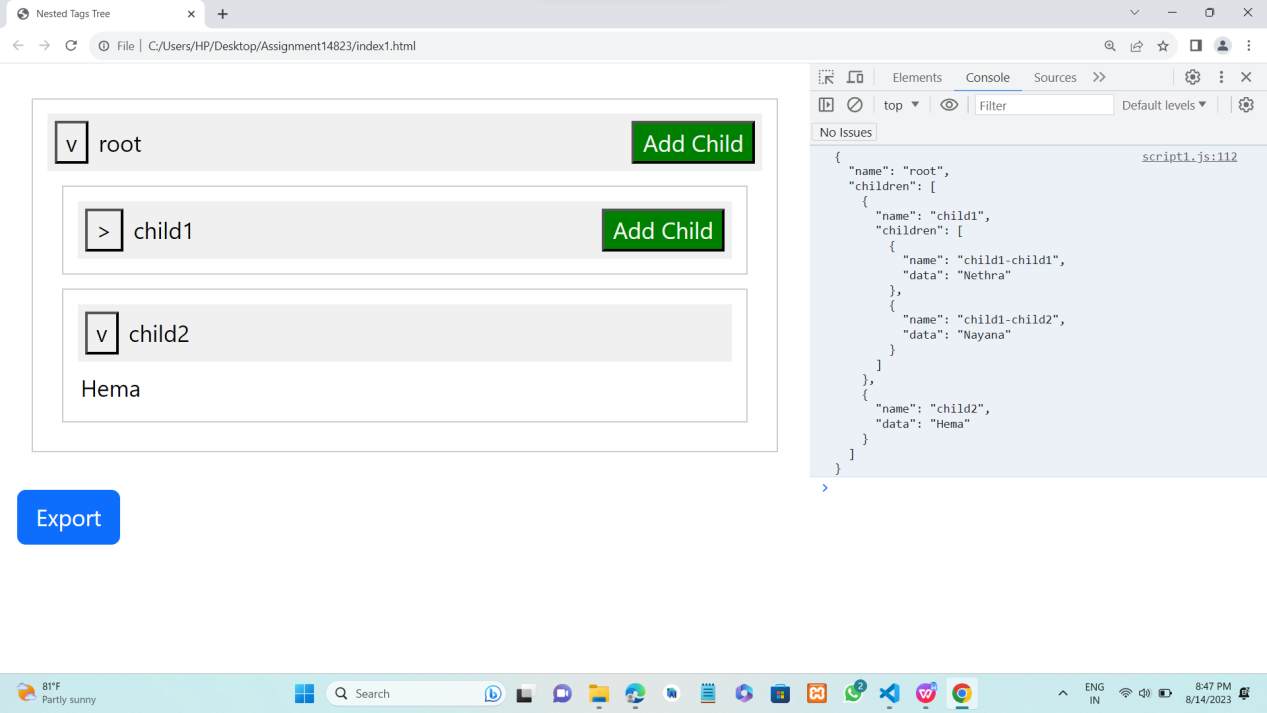
## Adding new child for “root”



**MINIMIZE :**



**MINIMIZE THE CHILD1 :**



Here I have given “AddChild” function only for “root” and “child1”, so “V” or “>” will work for these. BUT “V” or “>” will work for “chid2” and “child1-child1” and “child1-child2”.

